ECE 5582 Computer Vision

Lec 05: Interest Points Detection

Zhu Li
Dept of CSEE, UMKC
Office: FH560E, Email: lizhu@umkc.edu, Ph: x 2346.
http://l.web.umkc.edu/lizhu
Outline

• Recap of Lec 04
  • Image Filtering
  • Edge detection and HoG Feature

• About Homework-1

• Corner Detector
  • Harris Detector
Image Filtering

- 2D filtering: \([g] = \text{imfilter}(f, h)\)

- Properties:
  - Shift-Invariant:
    \(f(m-k, n-j) \ast h = g(m-k, n-j)\), if \(fh = g\)
  - Associative:
    \(f \ast h_1 \ast h_2 = f \ast (h_1 \ast h_2)\)
    this can save a lot of complexity
  - Distributive:
    \(f \ast h_1 + f \ast h_2 = f \ast (h_1 + h_2)\)
    useful in SIFT’s DoG filtering.
Derivative of Gaussian Edge Filter with direction

- Derivative of Gaussian as a smoothed edge detector: function of scale \( \sigma \) as well

- Any direction detection:

\[
\begin{align*}
\cos(\theta) &+ \sin(\theta) =
\end{align*}
\]
Hist of Oriented Gradients

- Generate local histogram per block

\[ f = (h_1^1, \ldots, h_9^1, h_1^2, \ldots, h_9^2, h_1^3, \ldots, h_9^3, h_1^4, \ldots, h_9^4) \]

<table>
<thead>
<tr>
<th>Angle</th>
<th>0</th>
<th>15</th>
<th>25</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
<td>15</td>
<td>25</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>45</td>
<td>95</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>47</td>
<td>97</td>
<td>10</td>
<td>12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Magnitude</th>
<th>5</th>
<th>20</th>
<th>20</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>10</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>30</td>
<td>30</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>70</td>
<td>70</td>
<td>80</td>
</tr>
</tbody>
</table>

Binary voting

Magnitude voting
HOG example

• HoG Feature:

```matlab
% compute HoG features
k=34; % dolphin
for j=1:4
    im = imread(sprintf('%s/%s/image_%04d.jpg', cal101_path, obj_name{k}, j));
    cellSize = 12;
    hog = vl_hog(single(rgb2gray(im)), cellSize, 'verbose', 'variant', 'dalaltriggs');
    hog_im = vl_hog('render', hog, 'verbose', 'variant', 'dalaltriggs');
    figure(11); subplot(2,2,j);
    imagesc(hog_im); colormap('gray');
    figure(12); subplot(2,2,j); imshow(im);
end
```
Outline

• Recap of Lec 04
  • Image Filtering
  • Edge detection and HoG Feature

• About Homework-1

• Corner Detector
  • Harris Detector
• Compute HSV color histogram for CIFAR 10 image retrieval
  • rgb2hsv
  • 10 class x 100 images each class
  • kmeans quantization of the HSV space
  • euclidean distance hist ranking
  • try other metrics, like KL distance as bonus
  • try spatio pooling as bonus.

• Compute Homography
SIFT Detection

```
[f1, d1] = vl_sift(im1); [~, n1] = size(f1);
[f2, d2] = vl_sift(im2); [~, n2] = size(f2);

% plot sift
figure(26);
n_sift=50; offs1 = randperm(n1);
offs2= randperm(n2);
subplot(2,2,1); imshow(im1c); hold on;
h1 = vl_plotframe(f1(:,offs1(1:n_sift))); set(h1,'color','r','linewidth',1);
subplot(2,2,2); imshow(im2c); hold on;
h2 = vl_plotframe(f2(:,offs2(1:n_sift))); set(h2,'color','m','linewidth',1);
input('n do matching...');
```

SIFT Match

```
[matches, scores] = vl_ubcmatch(d1, d2);
[sv, indx]=sort(scores, 'ascend');
match_offs = matches(:, indx);
offs1 = match_offs(1:m, 1);
offs2 = match_offs(1:m, 2);

% plot matching pairs
figure(26);
subplot(2,2,3); imshow(im1c); hold on;
h1 = vl_plotframe(f1(:,offs1));
set(h1,'color','r','linewidth',1);
subplot(2,2,4); imshow(im2c); hold on;
h2 = vl_plotframe(f2(:,offs2));
set(h2,'color','m','linewidth',1);
return;
```
• How it works?
• SIFT is indeed robust to affine transforms

% optionally, some affine transform
tform = maketform('affine',[1 0 0; .15 1 0; 0 0 1]);
im2c = imtransform(im2c,tform);
Outline

• Recap of Lec 04
  • Image Filtering
  • Edge detection and HoG Feature

• About Homework-1

• Corner Detector
  • Harris Detector
Interesting Points/patch: Use Case

- Find points/patch correspondence for robotics, image matching applications
  - Rotation, scale invariance – big challenge
Process of Interesting Points Matching

• Steps

1. Find a set of distinctive key-points

2. Define a region around each keypoint

3. Extract and normalize the region content

4. Compute a local descriptor from the normalized region

5. Match local descriptors

K. Grauman, & B. Leibe
Corner Detection: Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity

“flat” region: no change in all directions
“edge”: no change along the edge direction
“corner”: significant change in all directions

Source: A. Efros
• Key property: in the region around a corner, image gradient has two or more dominant directions

• Corners are repeatable and distinctive

How to match images?

- Find matching patches

Intuition: this would be a good patch for matching, since it is very distinctive (there is only one patch in the second frame that looks similar).
Corner Detection: Mathematics

- Window based detection:

  Change in appearance of window $w(x,y)$ for the shift $[u,v]$:

  $$E(u,v) = \sum_{x,y} w(x,y) \left[ I(x+u, y+v) - I(x, y) \right]^2$$
Corner Detection: Mathematics

• Residual energy as function of shift \((u, v)\):

\[
E(u, v) = \sum_{x,y} w(x, y) \left[ I(x+u, y+v) - I(x, y) \right]^2
\]

Change in appearance of window \(w(x,y)\) for the shift \([u,v]\):

\[
I(x, y)
\]

\[
E(u, v)
\]
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\):

\[
E(u, v) = \sum_{x,y} w(x, y) \left[ I(x+u, y+v) - I(x, y) \right]^2
\]

Window function

Shifted intensity

Intensit

Window function \( w(x,y) = \)

1 in window, 0 outside

Gaussian

DoG !

Source: R. Szeliski
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \)
for the shift \([u,v]\):

\[
E(u, v) = \sum_{x,y} w(x, y) \left[ I(x+u, y+v) - I(x, y) \right]^2
\]

We want to find out how this function behaves for
small shifts

\[
E(u, v)
\]
Approximate $E(u,v)$ with small change in $(u,v)$

$$E(u,v) = \sum_{x,y} w(x,y) \left[ I(x+u, y+v) - I(x, y) \right]^2$$

$$E(u,v) \approx E(0,0) + \begin{bmatrix} u \\ v \end{bmatrix} \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} u \\ v \end{bmatrix} \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

Local quadratic approximation of $E(u,v)$ in the neighborhood of $(0,0)$ is given by the second-order Taylor expansion:

Hessian
Corner Detection: Gory Details 😞

\[ E(u, v) = \sum_{x, y} w(x, y) \left[ I(x + u, y + v) - I(x, y) \right]^2 \]

Second-order Taylor expansion of \( E(u,v) \) about \((0,0)\):

\[
E(u, v) \approx E(0,0) + [u \ v] \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} [u \ v] \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix} [u \ v] \\
E_u(u, v) = \sum_{x, y} 2w(x, y)[I(x + u, y + v) - I(x, y)]I_x(x + u, y + v) \\
E_{uu}(u, v) = \sum_{x, y} 2w(x, y)I_x(x + u, y + v)I_x(x + u, y + v) \\
\quad + \sum_{x, y} 2w(x, y)[I(x + u, y + v) - I(x, y)]I_{xx}(x + u, y + v) \\
E_{uv}(u, v) = \sum_{x, y} 2w(x, y)I_y(x + u, y + v)I_x(x + u, y + v) \\
\quad + \sum_{x, y} 2w(x, y)[I(x + u, y + v) - I(x, y)]I_{xy}(x + u, y + v) \]
Taylor expansion of $E(u,v)$

$$E(u,v) = \sum_{x,y} w(x,y) [I(x+u, y+v) - I(x, y)]^2$$

Second-order Taylor expansion of $E(u,v)$ about $(0,0)$:

$$E(u,v) \approx E(0,0) + [u \ v] \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} [u \ v] \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{vu}(0,0) & E_{vv}(0,0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

- $E(0,0) = 0$
- $E_u(0,0) = 0$
- $E_v(0,0) = 0$
- $E_{uu}(0,0) = \sum_{x,y} 2w(x,y)I_x(x,y)I_x(x,y)$
- $E_{vv}(0,0) = \sum_{x,y} 2w(x,y)I_y(x,y)I_y(x,y)$
- $E_{uv}(0,0) = \sum_{x,y} 2w(x,y)I_x(x,y)I_y(x,y)$
Corner Detection: Gory Details

\[ E(u, v) = \sum_{x, y} w(x, y) \left[ I(x + u, y + v) - I(x, y) \right]^2 \]

Second-order Taylor expansion of \( E(u,v) \) about \((0,0)\):

\[
E(u, v) \approx [u \ v] \begin{bmatrix}
\sum_{x, y} w(x, y)I_x^2(x, y) & \sum_{x, y} w(x, y)I_x(x, y)I_y(x, y) \\
\sum_{x, y} w(x, y)I_x(x, y)I_y(x, y) & \sum_{x, y} w(x, y)I_y^2(x, y)
\end{bmatrix} [u \ v]
\]

\[
E(0,0) = 0 \\
E_u(0,0) = 0 \\
E_v(0,0) = 0 \\
E_{uu}(0,0) = \sum_{x, y} 2w(x, y)I_x(x, y)I_x(x, y) \\
E_{vv}(0,0) = \sum_{x, y} 2w(x, y)I_y(x, y)I_y(x, y) \\
E_{uv}(0,0) = \sum_{x, y} 2w(x, y)I_x(x, y)I_y(x, y)
\]
The quadratic approximation simplifies to

\[ M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]

where \( M \) is a second moment matrix computed from image derivatives:

\[ E(u, v) \approx [u \ v] \begin{bmatrix} u \\ v \end{bmatrix} \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I(\nabla I)^T \]
Corners as distinctive interest points

\[ M = \sum w(x, y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \]

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).

Notation:

\[ I_x \leftrightarrow \frac{\partial I}{\partial x} \quad I_y \leftrightarrow \frac{\partial I}{\partial y} \quad I_x I_y \leftrightarrow \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \]
The surface $E(u,v)$ is locally approximated by a quadratic form. Let's try to understand its shape.

$$E(u,v) \approx [u \ v] M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$
Consider a horizontal “slice” of $E(u, v)$:

$$[u \ v] \ M \ [u \ v]^T = \text{const}$$

This is the equation of an ellipse.

Diagonalization of $M$:

$$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \ R$$

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by $R$. The direction of the fastest change is determined by $\lambda_{\text{max}}^{-1/2}$ and the direction of the slowest change is determined by $\lambda_{\text{min}}^{-1/2}$. 
Interpreting the eigenvalues

- The eigenvalues of the Hessian matrix $M$:
  - Mix of “edge” and “corner” detection

Classification of image points using eigenvalues of $M$:

- $\lambda_1$ and $\lambda_2$ are small; $E$ is almost constant in all directions
- $\lambda_1 > \lambda_2$; “Edge” region
- $\lambda_1 \sim \lambda_2$; “Corner” region
- $\lambda_1 \gg \lambda_2$; “Flat” region

$\lambda_1$ and $\lambda_2$ are large, $E$ increases in all directions
Harris Corner response function

- Harris Detector:

\[ R = \text{det}(M) - \alpha \text{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha (\lambda_1 + \lambda_2)^2 \]

\[ \alpha: \text{constant (0.04 to 0.06)} \]
Harris Corner Detector

1) Compute $M$ matrix for each image window to get their \textit{cornerness} scores.

2) Find points whose surrounding window gave large corner response ($f >$ threshold)

3) Take the points of local maxima, i.e., perform non-maximum suppression

Harris Detector – Computing by Filtering

- Smoothed 2\textsuperscript{nd} moment matrix:
  - Apply Gaussian filter to smooth out noise:

\[
\mu(\sigma_I, \sigma_D) = g(\sigma_I) \ast \begin{bmatrix}
I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\
I_xI_y(\sigma_D) & I_y^2(\sigma_D)
\end{bmatrix}
\]

1. Image derivatives (optionally blur first)

\[
det M = \lambda_1\lambda_2 \\
\text{trace } M = \lambda_1 + \lambda_2
\]

2. Square of derivatives

3. Smoothed by Gaussian filter \(g(\sigma_I)\)

4. Harris Function – detect corners, no need to compute the Eigen values

\[
R_{\text{harris}} = \det[\mu(\sigma_I, \sigma_D)] - \alpha[\text{trace}(\mu(\sigma_I, \sigma_D))]^2 = \\
g(I_x^2)g(I_y^2) - [g(I_xI_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2
\]
Harris detector

- Input Images

- Compute Harris Response function: \( R_{\text{harris}}(x,y) = \det(M) - a \cdot \text{trace}(M) \)

Check out: testInterestPointDetector.m
Detection by Threshold on $R_{harris()}$

- Visually examine the points….
  - Repeatable points
  - Non-repeatable points…

$R_{harris}(x,y)$
Invariances: Affine intensity change

- Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$
- Intensity scaling: $I \rightarrow aI$

Partially invariant to affine intensity change
Invariances: translation

- Image Shift...

- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation
Invariances: Rotation

Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation
Corner location is NOT invariant to scaling!

All points will be classified as edges
Localization in Scale?
Automatic Scale Selection

• Function responses for increasing scale (scale signature)
Automatic Scale Selection

- Function responses for increasing scale (scale signature)
Automatic Scale Selection

- Function responses for increasing scale (scale signature)
Automatic Scale Selection

- Function responses for increasing scale (scale signature)
Automatic Scale Selection

• Function responses for increasing scale (scale signature)

\[
f(I_{k...l}(x, \sigma))
\]

\[
f(I_{k...l}(x', \sigma))
\]
Automatic Scale Selection

- Function responses for increasing scale (scale signature)
What Is A Useful Signature Function?

• Difference-of-Gaussian = “blob” detector
Difference-of-Gaussian (DoG)
Computation in Gaussian scale pyramid

- DoG filtering equivalent to difference of Gaussian blurred images

Distributive:
\[ f^*h1 + f^*h2 = f^*(h1+h2) \]
Find local maxima in position-scale space of Difference-of-Gaussian

Key point candidates from extrema detection

\[ L_{xx}(\sigma) + L_{yy}(\sigma) \Rightarrow List of (x, y, s) \]
Results: Difference-of-Gaussian

- DoG extremas
• Harris Detector
  • Harris Point Detection: Use 2\textsuperscript{nd} moment/Hessian matrix to estimate how repeatable the image patch is
  • Scale space invariance: gradient detection is a function of scale/sigma of Gaussian as well. Harris detector is NOT scale invariant
  • Interesting points description: invariance to rotation by rotating against dominant gradient directions. (SIFT scheme)